Information Protection

Curriculum Module SEI-CM-5-1.2 (Preliminary)

Information Protection

SEI Curriculum Module SEI-CM-5-1.2 (Preliminary)

July 1987

Fred Cohen
Lehigh University

Carnegie Mellon University
Software Engineering Institute

This work was sponsored by the U.S. Department of Defense.

This technical report was prepared for the

SEI Joint Program Office
ESD/AVS
Hanscom AFB, MA 01731

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

Revlew and Approval

This report has been reviewed and is approved for publication.

FOR THE COMMANDER

JOHN S. HERMAN, Capt, USAF
SEI Joint Program Office

This work is sponsored by the U.S. Department of Defense.

Information Protection

Contents

Information Protection

Module Revision History

Version 1.2 (July 1987)	format changes for title page and front matter
Version 1.1 (April 1987)	slight cosmetic changes
Version 1.0 (September 1986)	original version

Information Protection

Capsule Description

This module is a broad based introduction to information protection techniques. Topics include the history and present state of cryptography, operating system protection, network protection, fault tolerance and safety engineering, physical protection techniques, management tradeoffs, legal issues, and current research trends. The successful student in this course will be prepared for an in-depth course in any of these topics.

Philosophy

Given the impact of information on society, and the growing number of applications that require privacy and integrity in operation, it is important for software engineers to be aware of the available techniques and policies for information protection, and their advantages and disadvantages in a wide variety of situations. This module is designed as a broadly based introductin to information protection concepts applicable to the work of any software engineer, and as such, it has primary responsibility for introducing some of the problems, solutions, and pitfalls of protection systems. If there is a single concept that the student should learn from the module, it is that the design and implementation of sound protection systems are extremely complex, and the ramifications of protection system failure can be extreme.

This module does not provide in-depth coverage. For certain audiences, more detailed coverage of some or all topics may be appropriate. The bibliography provides references to sources of additional material that may be added at the instructor's discretion.

Objectives

Cryptography. The student should be able to demonstrate several encryption techniques, methods by which they may be attacked, and estimates of the time required to attack them successfully under reasonable assumptions. The use of cryptosystems for the protection of information should be well understood, and the concepts of authentication and encryption should be clearly differentiable. Some simple private and public key protocols and design trade-offs, in terms of space and time, should be demonstrable. The statistical nature of language should be clearly understood as a critical key in the analysis of most cryptosystems. A healthy skepticism about products on the market should be demonstrated, and a clear understanding that cryptosystem design is difficult should be displayed.

Operating System Protection. The student should be able to identify a number of different attacks against operating systems and to indicate which are relatively easy to prevent and which are virtually impossible. The student should be familiar with a number of policies for the protection of information, and demonstrate a clear understanding of the place of protection policy in the design of protection systems. The student should be able to differentiate between attacks that are preventable by proper implementation, those which require protection policy measures for their limitation, and those for which there is no known defense. A broad knowledge of protection models, including the subject object model, the security and integrity models, the lattice and poset (partially ordered set) models, and the difference between identification, authentication, and authorization should be shown. The student should understand in depth the concept of defense and be able to explain the rationale behind it. Students should be able to carry on an intelligent discussion with regard to the trusted system evaluation criterion and its role in operating system protection. The role of audit, testing, and life cycle assurance should be clearly

understood. The student should be able to explain the simplifications of general operating system protection which apply to database protection, demonstrate a clear knowledge of database protection problems, and demonstrate examples of the data agregation problem. The difficulties involved in eliminating this problem should be clearly understood, and the complexity of the problem should be demonstrable. Business and social implications of database technology should be clear.

Network Protection. The student should be able to identify the two opposing schools of thought on the design of network protection, and be able to identify the similarities between networks and operating systems in terms of protection policies, models, and implementations. The role of cryptography in network communication and the need for authentication in a network protection system should be explainable. The student should be able to identify and explain, in a general way, the covert channel and traffic analysis problems and why they are difficult to deal with. Given the architecture of a network, the student should be able to point out aspects that lend themselves to protection or violation of protection, and explain some of the ramifications of changes in the structure of the network.

Fault Tolerant Computing and Safety Engineering. The student should be able to explain the need for fault tolerant computing and safety techniques and be able to identify those systems where such techniques are usually most critical. The nature of faults that lead to computational errors and safety problems should be explainable, and a number of techniques for reducing the likelihood of these problems should be demonstrable. The difficult problems in fault tolerance and safety should be clearly understood, as should be the concept of risk analysis.

Physical Protection. The student should be able to explain the relevance of physical information protection measures for software protection, and should be able to generate possible software solutions to simple physical protection problems. The rationale behind tempest protection and the need for and limitation of backups should be clearly understood.

Management Issues. The student should demonstrate a clear understanding of at least one method of risk analysis and one means of performing cost benefit analysis in a protection-relevant setting. The problems with determining the probability of attack should be clear, and the effect of errors in this domain should be understood in a broad sense. The role and problems of background checks, security clearances, and other similar measures in the quest for protection should be displayed. Other management issues should be clearly discernable, and techniques in common use should be demonstrable at a simple level.

Legal Issues. The students should demonstrate an awareness of social aspects and effects of protection systems, and the potential ramifications of the design and implementation of such systems in various environments. The should be able to discuss several ethical and moral issues from at least two opposing viewpoints. The likely effects of a variety of protection systems on personal computers, and the effect of secrecy on research and development should be clear. A working knowledge of the laws which affect the software engineer in regard to the violation of protection systems, copyrights, patents, trade secrets, and the bases upon which they have historically been formed should be understood.

Prerequisite Knowledge

There are no explicit prerequisites for this course, but it is expected that students will have a programming experience, working knowledge of engineering mathematics, and an inquisitive mind. For in-depth coverage of various areas within this module, considerably more background is required.

Module Content

Outline

1. Introduction and Overview
2. Cryptography
   a. Basics
   b. The Pre-Computer Age
   c. Information Theoretic Cryptography
   d. Private and Public Key Cryptography
   e. Cryptographic Protocols
   f. Summary and Systems on the Market
3. Operating System Protection
   a. Basics
   b. Attack and Defense
   c. Protection Policy
   d. Protection Models
   e. Implementation Techniques
   f. Standards: The Trusted System Evaluation Criterion
   g. Examples
   h. Database Protection
4. Network Protection
   a. Theoretical Basics
   b. Policies and Models
   c. Implementations
   d. Standards
   e. Examples
5. Fault Tolerant Computing and Safety Engineering
   a. Basic Concepts
   b. Policies and Models
   c. Analysis Techniques
   d. Specific Protection Techniques
6. Physical Protection
   a. Military Tempest Requirements
   b. Prevention of Physical Attacks
   c. Detection of Physical Attacks
   d. Backups and Facilities Planning
7. Protection Management
   a. Asset Identification and Valuation
   b. Probability of Attack
   c. Expected Loss
   d. Analytical Techniques
   e. Personnel Techniques
8. Legal Issues
   a. Security Agencies and Departments in the US
   b. Moral and Ethical Considerations
   c. Legal Considerations
9. Current Research Trends

Annotated Outline

I. Introduction and Overview

The module introduction will acquaint the student with the concept of information protection in light of the current information revolution. Simple social, legal, business, personal, and educational situations will be presented to point out the wide reaching implications of information protection problems, and to point out the need to understand the basic philosophical, theoretical, and technological aspects of the field. The introduction can be presented at a level that is easily understood by any educated person, in the same manner as an after dinner speech. Several critical and seemingly unsolvable protection problems should be introduced as solvable under present technology (their solutions, of course, will be presented later.)

II. Cryptography

1. Basics

Cryptography is introduced as a means of transforming information into a form in which its meaning is obscured. Extremely simple examples are given (e.g., igpay atinlay) and shown to obscure information from a small segment of the class for a short period of time. The problem of obscuring information from most people for a great deal of time is then introduced, and its implications for the protection of information are pointed out. Two major implications that should be introduced are the use of cryptography for secrecy and the use of cryptography for authentication. These concepts should be expanded to the protection of information from illicit dissemination and modification respectively, and these concepts should be further expanded and explained with examples. The one key, two key, and two lock box cryptosystems should be introduced, and the concept of active and passive tappers in a communication system should be shown. Graphic aides for these presentations should be provided.

a. Obscuring meaning

b. Secret languages

c. Secrecy and authentication

d. Locks and boxes

e. A simple communications system

f. Active and passive tappers

2. The Pre-Computer Age

The history of cryptography should be introduced with the fact that cryptosystems have been in use for several thousand years, and that in all that time, only one cryptosystem designed prior to 1970 is still unbroken. We can conclude from this that the design and use of cryptosystems is not a trivial matter, and that if there is any hope of designing practical and secure cryptosystems, we must learn how and why they work and how and why they are broken. Examples should be cited from Rome where senators would be sent out to deliver enciphered messages to generals, the messages often containing the message 'Kill the bearer of this message,' or other similar cases where lives have been lost or saved by code making and code breaking. Many references to these types of cases exist in public libraries under the subject heading of cryptography. The breaking of the Enigma machine, the decision to keep it secret, the use of radar as a cover, and the trade-off in terms of lives in the bombing of London should be used to exemplify the nature, criticality, and ethical concerns involved in this field.

The Skytail cipher, the Polybius square, Caesar cipher, wheel cipher, and the Playfair system should all be introduced and demonstrated with simple examples. These are commonly found in public libraries under the subject heading of cryptography. A set of demonstration wheels is most useful in the demonstration of the Caesar and wheel ciphers. Each of these systems should be broken by purely statistical methods, and the demonstration wheels should be used extensively to show the statistical nature of language. A typical example is the demonstration that the wheel cipher is unlikely to come up with two plaintext words simultaneously; the computation of the probability of such an occurrence should be calculated in class. References to this material are widely distributed, and can be found in [Denning82] among others.

A classification scheme for describing the space of cryptosystems should be included in a handout and described in detail along with examples of each type of cipher and a simple cryptanalysis problem [Knight78]. The use of automated tools for breaking these cryptosystems should be introduced, and a sample tool of this sort should be provided with the course. Famous unsolved ciphers should be cited, and perhaps a famous puzzle or two should be shown. Assignments for this portion of the course should include breaking several small ciphers with and without the use of computers.

a. Ages of broken ciphers

b. Staff ciphers

c. Caesar ciphers

d. Jefferson wheel ciphers

e. Playfair cipher

f. Automatic cryptanalysis assistants

g. A classification of cryptosystems

h. Unknown languages

i. Famous unsolved ciphers

j. Cryptoanalysis of ciphers

3. Information Theoretic Cryptography

Shannon's work in cryptography should be introduced as the first mathematical investigation of cryptography, and the basis for virtually all of the work since then [Shannon49]. The statistical nature of language should be presented in the manner Shannon presented it, and some extensions of Shannon's work, such as the derivation of the index of coincidence, should be presented [Knight78]. The monkey at the keyboard analogy should also be presented as an alternative demonstration of the effect of statistics on the use of language, and a demonstration program should be provided to facilitate this.

The *unicity distance* should be derived from the statistical nature of language [Shannon49]. These effects should be used to demonstrate cryptanalytic techniques against many of the classes of ciphers presented in the classification scheme. The index of coincidence, known ciphertext, known plaintext, assumed word, and other similar techniques should be demonstrated [Knight78].

Diffusion and confusion should be introduced, and the concept of workload in the breaking of cryptosystems should be presented. The perfect cipher should be introduced and shown to have an infinite unicity distance [Shannon49]. The use of data compression to reduce linguistic redundancy should be demonstrated to increase the unicity distance as well [Chaum85], and a demonstration of a perfect cryptosystem with a finite length key in a statistically independent language through the use of data compression should be shown. The practical limitations of this scheme should be noted. Examples of diffusion and confusion should be given, and these should be used to introduce the Data Encryption Standard (DES) cryptosystem [NBS77].

The DES should be examined in terms of its successive rounds of diffusion and confusion, with a trade-

off involving the infeasibility of doing full diffusion or confusion over the entire plaintext block. The structural design of the DES should be shown to reflect this design trade-off, and the elements of diffusion and confusion should be made clear [Kam79]. The derivation of numbers for the transforms in the DES and the limited key length [Diffie77] should be pointed out as questionable in terms of the actual security of the system, and a case should be presented for the questionable security of the system. A good assignment at this point would be the division of the class into two groups, one to defend the DES and the other to attack it. Several good papers on the subject can be used as the basis for this discussion. More in-depth discussion in this area requires a solid understanding of number theory and computational complexity, and the recent research should be used as a guide if this topic is to be pursued further.

The use of the Enigma machine in World War II should be cited as an example of a diffusion and confusion cipher, and a description of the breaking of this famous system and the manner in which the British used it should be presented and discussed. References to this material exist in most public libraries under "cryptography."

a. Statistical nature of language

b. Example cryptanalytic techniques

c. Unicity distance

d. The workload coefficient

e. Perfect ciphers

f. Data compression effects

g. Confusion and diffusion

4. Private and Public Key Cryptography

The concept of public key cryptosystems should be introduced, and classical papers should be used to show the development of the concept [Diffie76]. The analogy to the puzzle problem is strong here, and should be used to explain each of the subsequent systems. The workload concept should be shown to be reflected in public key systems, in that these systems are entirely workload dependent with unicity distance zero. The development of two cryptosystems based on the public key concept, knapsack and RSA (named for Rivest, Shamir, and Adleman; see [Rivest78]), should be given in gross terms; the making of both systems and the breaking of the knapsack should be explained [Rivest78, Adleman82]. Extensions of the RSA should be noted as prevalent in the literature [Chaum83b, Chaum84]. Detailed explanations of why and how the RSA and knapsack work and why and how the knapsack can be broken require significant understanding of number theory, computational complexity, and recent advances in cryptography. These topics tend to be too difficult for most students; the ramifications of the technology, rather than the details of its operation, should be explored here.

a. The Data Encryption Standard

b. The puzzle problem

c. The knapsack's creation and destruction

d. The RSA

5. Cryptographic Protocols

Protocols for cryptosystems should be introduced in terms of the birth problem, the key distribution problem, and the active and passive tapper problems. Public key cryptosystems should be shown to solve these problems. A case study should be used to point out a state-of-the-art use of a public key protocol, and we suggest the nuclear test ban treaty verification problem [Simmons81]. The new field of cryptographic protocol analysis should be introduced, and basic principles should be given. In-depth understanding of cryptographic protocols requires expertise in number theory, logic, finite state machines, program proofs, and related fields. Cryptographic protocol analysis is so new that we can only say that it will be important in information networks where cryptographic protection must be proven.

The performance issues involved with the DES and RSA should be introduced, and design trade-offs pointed out [Rivest78, Kam79]. The number of keys required for a private key system should be shown to grow as n^2 in the number of communicating parties, while it grows as 2n in a public key system. The advantages for networks should be made clear. Multiple encryption and decryption with the same technique should be shown to be of questionable value in the security of a cryptosystem, while the use of multiple techniques should be shown to have potential but unproven advantages. The use of hybrid private/public key systems for eliminating the key distribution problem while maintaining acceptable performance should also be shown. In-depth understanding of these issues requires detailed understanding of the mathematical properties of cryptosystems.

Several systems on the market should be introduced and where possible, techniques for attacking them should be demonstrated [Highland84]. In summary, it should be pointed out that history shows cryptosystems to be complex and fragile mechanisms, and that historically, cryptosystems have eventually been broken. The casual design of cryptosystems should be heavily discouraged, and trust in a cryptosystem should be questioned at every turn. Students should be taught to view with extreme skepticism any cryptosystem on the market, and to seek out expert assistance before implementing cryptography in a system

or network.

a. Key distribution

b. Protocols

c. The key distribution problem

d. Key maintenance problems

e. Performance problems

f. Hybrid systems

6. Summary and Systems on the Market

III. Operating System Protection

1. Basics

Operating systems are introduced as a means for allocating resources. Operating system protection is introduced as a means for protecting resources while allocating them. In order to understand how to protect resources, we must first understand what we mean by protection.

2. Attack and Defense

The concept of attack and defense should be introduced with specific examples, including exceptions, covert channels, traffic analysis, line tapping, spoofing, data diddling, Trojan horses, computer viruses, other transitive informational attacks, and denial of services attacks [Lampson73, Shoch82, Cohen84b, Linde75]. Examples of each of these should be demonstrated, and the ramifications of these attacks on information systems should be explained. Example attacks and defenses should be used to demonstrate the ramifications of policy decisions. Some of the advanced attacks, such as traffic analysis and computer viruses, require significant time to explain fully, and their detailed explanation may be left to more advanced courses if time does not permit their exploration here.

Denial of services should be explored in terms of availability analysis [Siewiorek82], and the impossibility of ideal defenses should be pointed out as specific cases of undecidable problems. The role of the operating system in protection from denial of services should be introduced as a resource allocation problem, and the trade-offs involved in attack and defense should be pointed out. The use of denial of services for covert channel transmission should be pointed out, and an example of a covert channel involving denial of services on disk and in CPU usage should be used to exemplify that the denial of service problem is at least as difficult as the covert channel problem. In-depth understanding requires some information theory; in particular, the results that a signal can be successfully sent regardless of the amount of noise in the channel, and that a reduction of bandwidth can be achieved with an increase of noise in the channel.

a. Exception attacks

b. Covert channels and traffic analysis

c. Line tapping

d. Spoofing

e. Data diddling

f. Trojan horses

g. Computer viruses

h. Other transitive attacks

i. Denial of services

3. Protection Policy

The highest level of policy planning produces a *protection policy* that embodies our policy intentions with regard to the protection of information. Effects of inadequate, inconsistent, or incomplete policies should be demonstrated with simple examples [Klein83]. The reflection of desired goals in this policy should be introduced, and the technical feasibility of these goals should be given as the reason for our technical examination. A number of policies should be examined, including the security and integrity policies, the lattice and poset policies [Cohen86], Harrison's protection paper [Harrison76], identification, authentication, and authorization [Denning82], auditability and auditing techniques, defense in depth, and other theories. Graphics should be provided for each of these policies, and explanations of strengths and weaknesses and their combinability should be included. A detailed understanding of policy is complex, and the potential ramifications of decisions may be difficult to predict. Examples may be more important than detailed understanding, the point being to indicate the difficulty in producing an appropriate policy.

a. Security model

b. Integrity model

c. Lattices

d. Posets

e. Harrison's protection paper

f. Identification

g. Authentication

h. Authorization

i. Audit

j. Defense in depth

k. Other techniques

4. Protection Models

The use of a formal *protection model* that models the resources in an information system in a manner that is sufficiently formal to allow in-depth mathematical analysis of algorithms and protection properties should be introduced as the next step towards implementation and understanding. This is

often referred to as a *formal top level specification* (FTLS), and is used to derive implementation [Klein83]. An example of such a specification and the clear mapping between a policy and that specification is worth showing, but may be time-consuming. Detailed analysis requires a great deal of expertise in the theory of program correctness, and a clear understanding of finite state automata.

5. Implementation Techniques

A number of modeling and implementation techniques should be examined, including the concepts of a security kernel, subject/object matrices, flow control matrices, access lists, capabilities, proof of correctness, and other techniques. At least one simple example of each system should be given, and a sample model should be shown to reflect a specific policy [Denning82]. Modern algebra, the theory of program correctness, and finite state automata theory are required for in-depth understanding. At the introductory level, simple examples are most useful.

Implementation of formal specifications should be covered in terms of the techniques required to correctly generate, demonstrate, and test the correctness of an implementation. This should include execution time, compile time, and verification based flow control, proof of correctness, and verification [Denning82]. At least one simple example should actually be proven correct by use of finite state machine model [Benzel84]. Examples should be extended into the hardware design area, and an example system based on a hardware rather than software implementation should be shown. A covert channel example should be given in light of this design, and the bandwidth of this channel should be determined given a simple set of assumptions. This may be too deep for many students, and should be regarded as optional. It drives home the point that the design of protection systems is not simple and straightforward.

a. Security kernels

b. Subject/object matrices

c. Flow control matrices

d. Access lists

e. Capabilities

f. Formal proof techniques

6. Standards: The Trusted System Evaluation Criterion

The *Trusted Systems Evaluation Criterion* (TSEC) should be introduced as a guidepost in understanding operating system protection [Klein83]. Testing and assurance, auditability, administration, and other related protection areas should be covered in some depth, with the understanding conveyed that theory is available in the literature. The criterion should be reviewed in some detail with special attention to the graphic depiction of the criterion as a whole. In-depth understanding of the criterion requires a great deal of related material, and is often only clarified by discussion with those who wrote it. Much of it has not even been clarified by the writers. It should be pointed out that this is just a beginning in the search for standards.

7. Examples

Example systems should be given from the literature, and the results of TSEC evaluations should be shown for all known systems. The ramifications of these results should be clearly pointed out, and examples should be used from the nuclear missile and missile defense system perspective to point out the potential hazards involved in faulty operating system protection [McCauley79, Feiertag79, Popek79]. Integrity corruption as well as security leaks should be cited in the examples. Security by obscurity should be pointed out as the classic example of how not to build a protection system. Detailing the evaluations given to systems in terms of the simple diagram from the criterion will serve to tell the students precisely what they can and cannot expect from a system evaluated at a given level.

a. PSOS

b. KSOS

c. UCLA Secure UNIX

d. MULTICS

e. SCOMP

8. Database Protection

Database protection should be introduced as a special case of operating system protection in which there is limited functionality. The ramifications that databases already have on individual privacy and freedom should be introduced, and examples should be used liberally, such as the TRW credit database and the effects of attacks against it, the national medical database and its regulations preventing individuals from attaining information about themselves, and databases spread throughout the government, whose information may or may not be available to the individual. The problems of these databases should be balanced with their advantages; for example, quick and accurate credit ratings, immediate verification of personal data, and the protection and consolidation of information on public record.

The objective of database protection is to ensure the secrecy, integrity, and availability of data stored in the database. The database protection problem may be viewed like that of operating system protection, except in that functionality is often special purpose rather than general purpose. Therefore, the general principles and techniques of operating system pro-

tection apply, but may be augmented because of the further specificity of the database [Fernandez81].

The use of a subject/object model for controlling access, and the effects of transitivity should be pointed out. An example of a transitive database attack, wherein a check is generated through the misentry of an incoming payment or a similar attack, should be demonstrated to clarify the potential complexity of the protection problem.

Statistical attacks on databases may be regarded in the same manner as covert channels in computer systems, as unintended transmission of signals through noisy channels. An example of several statistical attacks and defenses should be used to demonstrate the complexity of the protection problem. The underlying mathematical problems should be reviewed, and the complexity of defending against attacks of this sort should be pointed out, with examples from papers in the field [Denning82].

Field encryption techniques, statistical summary techniques and attacks, information reduction, and noise introduction should be discussed, and analogies to the covert channel problem should be clearly exemplified.

Alternative security, integrity, and availability models, including those used in the section on operating system protection, should be introduced and explored, research potentials should be introduced, and classroom discussion should be encouraged. The level of coverage should be the same as in the previous sections because the technical aspects of this field depend on understanding network and operating system protection. In-depth coverage of the aggregation problem depends only on thorough understanding of completeness, Turing capability, and logic. It should be explained by example and understood by all. The complexity issues in defending against such an attack require considerably more expertise. This topic is well covered in [Fernandez81].

a. Basic Concepts

b. Policies and Models

c. Data Aggregation Effects

d. Complexity Issues

IV. Network Protection

Protection in computer networks is similar to protection in computer systems, except that the distribution of hardware and facilities makes communications between resources less trustworthy, and the potential for compromise of individual sites affecting the entire network is made substantial. Since the field of network protection is new, there is no single dominant philosophy upon which to base our work. There are, however, some widely considered problems and a number of potential solutions to them.

In-depth understanding of network protection depends heavily on in-depth understanding of cryptography and operating system protection, and as such should be covered in the same degree of depth, if time allows.

1. Theoretical Basics

The two basic philosophies are:

- A network is just like a computer and we should use the same standards.
- A network consists of a set of computers and a communications system.

The former perspective takes the theoretical viewpoint that a network is a system like any other system, while the latter considers the practical aspects of protecting communications lines. These philosophies should be explored and contrasted for their philosophical and practical values, and other view points should be discussed [Walker84].

2. Policies and Models

As a basic principle, we consider a network as a means for making several physically distributed facilities appear to the user as a single facility with enhanced capabilities. If we are interested in the protection of information within a computer system, we tend to control the information flow, and it seems logical to extend this to computer networks. The distributed domains in computer networks should be explained as a logical extension of the domains used in computer systems. The protection of communication should be considered in terms of secure and insecure channels, and methods by which insecure channels may be made secure [Cohen85, Cohen86].

Basic communications network architectures and topologies should be explored for their inherent vulnerabilities and protection capabilities. In particular, the problems with Ethernet type networks and satellite links should be examined in some depth, and their inherent vulnerabilities exemplified.

a. Distributed domains

b. Communications protection

c. Basic communications networks

3. Implementations

The traffic analysis and covert channel problems should be explained in some depth, and their unsolvability should be reiterated. Information theoretic techniques to reduce covert channel and traffic analysis bandwidth should be reviewed. Analysis of attacks on networks under various takeover assumptions should be carried out, and the mathematical basis for takeover and attack analysis should be given [Cohen85, Cohen86].

The application of cryptography for protecting information channels in a computer network should be pointed out and explored in terms of the concept handled in the cryptography portion of the course. The application of techniques from the operating system protection part of the course should also be explored in terms of their utility in understanding and designing secure computer networks.

a. Traffic analysis

b. Takeover analysis

4. Standards

The lack of standards for network security should be pointed out, with the current development process of both the *Trusted Network Evaluation Criterion*, and the open system interconnect (OSI) network standard of the International Standards Organization (ISO). These standards should be reviewed in some depth, and known vulnerabilities pointed out. Current research trends should also be examined to show the state of the art.

a. The Trusted Network Evaluation Criterion

b. The OSI standard

5. Examples

V. Fault Tolerant Computing and Safety Engineering

Fault tolerant computing should be introduced as a means to the storage and use of information in the presence of physical faults. Safety engineering should be explained in terms of preventing physical harm caused by inadequacies in the coverage of faults. Examples of deaths, near misses, and errors found in simulation should be used to point out the criticality of these issues. The tradeoffs among fault tolerance techniques, safety techniques, and system costs should be discussed and demonstrated with examples. Management techniques for software safety should also be discussed.

The objective of safety engineering is to attain "freedom from those conditions that can cause death, injury, occupational illness, or damage to or loss of equipment or property." To do this, we eliminate hazards or reduce their likelihood to an acceptable level. When assessing the acceptability of risks, we must take a rational approach. A typical method for assessing risk is to compare the risks of the system under investigation to the risks incurred in everyday life. It is typical to find that risk standards (those for nuclear power as an example) require that risk be reduced to levels well below the risks taken in everyday activities (in terms of the average reduction in life expectancy). The irrationality of spending great deals of money to reduce risks by an inconsequential amount in one application when spending the same amount in another application would save a great many more lives should be pointed out with specific policy examples currently in use.

Most safety work is closely related to the field of fault tolerant computing, and a knowledge of hardware techniques is most helpful in software safety. The complexity associated with proof of correctness and thorough testing should be examined and shown to make provably safe systems infeasible for most nontrivial problem solutions. The close relationship between assurance in operating systems and safety systems should be explained and exemplified.

The use of policy, models, and design techniques analogous to the design of secure operating systems should be examined, noting particularly that safety issues are not as clear or well developed as those for operating systems, and that the eventual future of safety engineering is intimately tied to the ability to design safe operating systems.

Typical techniques, such as hazard identification, fault analysis, and standardization, should be discussed in terms of their utility in practical situations. Intrinsic safety, hazard control, and warning systems should be explained, and techniques to affect each of these should be described in some detail. Lockouts, lockins, interlocks, timeouts, sanity checks, built-in self test, certification, simulation, and other related techniques should be given as examples, with papers from the literature used to supplement the course where appropriate.

1. Basic Concepts

2. Policies and Models

3. Analysis Techniques

4. Specific Protection Techniques

VI. Physical Protection

Although physical security is not strictly a software engineering issue, the ramifications of physical attacks and errors, and the potential for software solutions to these hardware problems should be explored. Furthermore, it should be made clear that regardless of the attempts of the software engineer to protect a system, there are physical factors involved in the protection of information. The use of assumptions with regard to the scope of the software problem should be pointed out in terms of the tradeoff between hardware and software solutions.

An in-depth understanding of this material requires considerable understanding of computer engineering, communications, information theory, and the state of the art in signal detection techniques. This module should only cover this material in a broad sense, and should emphasize the nature of the problems rather than detail the methods. The emphasis should be kept on the effects on software engineering.

1. Military Tempest Requirements

The concept of electronic signal detection and exploitation should be introduced, and the ramifications of tempest attacks should be made clear. Military tempest requirements should be examined at a surface level, with special attention paid to the expenses and complexities involved in physical solutions, and the ramifications of these costs to the software domain.

2. Prevention of Physical Attacks

Software solutions to physical attacks should be explored, including the use of cryptography to nullify the gain of hardware violations, the detection and correction of hardware failures with software, redundant software to circumvent case dependent hardware failures and transient faults, and the use of software to counter denial of services attacks.

3. Detection of Physical Attacks

4. Backups and Facilities Planning

The role of backups and facilities planning in the protection of information systems should be explored, with special attention to the potential abuse of redundancy in the protection domain, the limitations of backups for integrity protection, and the effects of distribution, repair, and maintenance procedures on information protection. Scheduling strategies for backups, and techniques to remind users of the necessity of backups, should be discussed in detail. Hot standbys, cold standbys, redundant processors, coding techniques, and other aspects of fault tolerant computing should be mentioned as important areas.

VII. Protection Management

The use of protection mechanisms has associated costs and personnel, and the lack of protection also has costs. In the engineering of software systems for the protection from informational harm, it is important to understand the economic and personnel ramifications of protection decisions. Examples should be used to point out the potential loss due to integrity corruption, information dissemination, and denial of services in specific situations. Situations, such as a corruption that causes a lawyer to miss a court date, or medical systems that give wrong diagnoses, should be used as examples. Potential harm in terms of lost investor and customer confidence should also be considered and discussed.

In-depth understanding of this field depends on in-depth understanding of risk analysis and planning, and this material should only be covered in broad form. At least one example should be given of a complete analysis of a simple system.

1. Asset Identification and Valuation

Techniques for asset identification and valuation, including audit, inventory, exposure determination, insurance investigation, and other methods, should be discussed. The U.S. military model of asset identification and valuation should be stressed as a realistic technique in regular use. The identification of losses due to integrity corruption, illicit dissemination, and denial of services should be exemplified, and the use of all three in analysis should be stressed.

2. Probability of Attack

The estimate of attack probabilities is a key aspect of risk analysis, and there is no universally accepted predictive method. A number of methods should be examined for their good and bad qualities, and the engineering nature of the choice of method should be stressed.

3. Expected Loss

The computation of expected loss per unit time should be performed using straightforward techniques, and an example problem should be worked through in its entirety.

4. Analytical Techniques

Analytical techniques, such as sensitivity analysis, hill climbing for optimizations of resource utilization, and other techniques from cost/benefit analysis, should be examined as decision making methods in protection system implementation. Special emphasis should be placed on the inexact nature of these analyses because of the human factors involved, and the inadequacy of past history to accurately predict future behavior. The ongoing nature of this analysis throughout the systems lifetime should be examined and further discussion solicited. Lifecycle costs should be mentioned as a current trend in this sort of analysis.

5. Personnel Techniques

Techniques for performing and attaining personnel checks, hiring and firing, observing personnel as they operate, and managing people with regard to protection issues should be discussed in some breadth. Common educational and training techniques, methods such as handwriting analysis, background checks, and techniques for covering attacks by internal personnel should be introduced.

VIII. Legal Issues

It is important for engineers to understand some of the social ramifications of their actions, and to maintain some degree of social responsibility. The motivations, ethics, and morals of individuals vary widely, and it serves a useful social and technical purpose to explore these areas, especially in terms of the vast social and technical responsibility of software engineers in an information based society. The legal system has become increasingly strained because of the social ramifica-

tions of the information systems now in place, and this situation has brought about legal decisions that seem to have little basis in technical reality. In order to come to grips with these problems, it is important to understand the social situations that bring about the legal decisions, and the historical basis for legal determinations.

1. Security Agencies and Departments in the US

   Security agencies and departments in the United States should be introduced, and their respective functions described. The CIA, FBI, DOJ, NSA, DOD, and DOT should be described in terms of their missions, past successes and failures, and historical significance. Their respective roles in the protection of information should be outlined and discussed.

2. Moral and Ethical Considerations

   Moral and ethical issues of plagiarism, giving false information, storing false information, and giving away information about others should be discussed.

   The Freedom of Information Act has special import at the federal level, and each government agency is required to maintain specific standards of information control. The Privacy Act of 1974 guarantees the rights of individuals and corporations from information problems. Both should be examined.

3. Legal Considerations

   Legal considerations often dominate the lives of software engineers after they produce products for the marketplace. Many protection systems have been used for preventing illicit copying of microcomputer software, and several companies have lost major law suits as a result of their use of abusive protection mechanisms [Voelcker86]. A brief review of legal motivations [Tompkins84] and legislation [Voelcker86] in information protection should be used to increase the student's awareness of legal issues in information protection.

   The student should understand that there is a fundamental difference between legal proof and scientific proof. Legal proof in the United States is any argument that would convince 6-12 average Americans on the street, whereas scientific proof is somewhat more explicit in its form and content. It is important for students to understand that the law should be viewed tactically rather than contemptuously, and that actions and the intent of actions should be well documented in order to support their position.

   Fundamental concepts are introduced, such as computer systems as property, and the analogy to trespassing, illegal entry, and illegal search and seizure; wiretapping laws; programs that punish illicit users by violating their property, and the analogy to a shotgun pointed at your door that goes off when entered; electromagnetic emanations, and the principle that if it doesn't disturb you or cause you harm, you have no right to prevent it; software licenses vs. sales, patent vs. copyright vs. trade secrets; reverse engineering, broken seal legal agreements, freeware, transitive copyrights, and derived works; legality of sales to outside nations (cryptographic equipment especially), legality of sending encrypted messages; implied warranty of merchantability: does it apply, can it be waived, what right does it give the consumer to violate protection if the product is not worthwhile, do you have a right to modify for your own use.

   Many of these aspects have yet to be well defined or tried in court, and the tenuous nature of these issues should be explained to the student. The instructor should seek to give a broad picture and perhaps a few examples where there is considerable controversy over the prevailing legal opinion. Basic principles of criminal vs. civil law should be introduced, and the concepts of legal proof explained.

   The legal basics of copyright, patent, trade secrets, and other legally protected forms should be introduced and explained. The historical basis for these protections should be derived from the earliest uses of copyrights to protect governments from public scrutiny through the protections provided by the U.S. Constitution, and legal principles as they have developed. Recent legal decisions and laws should be reviewed with a particular eye toward the trends of the times and their effects on the development of the law.

IX. Current Research Trends

A review of course material is often aided by the investigation of selected papers in the current literature. A technique that has proven fruitful in the past is to set each student on a research project to read about and present a recent research result in light of the information learned in the class. This exercise serves as a test of the students understanding of the material and as a method for generating more specific interest for further course work and research.

Teaching Considerations

Suggested Schedule

The following sample class plan is based on the offering of 15 one-hour lectures with outside lab time and homework time, and exams outside of class time.

1. Introduction (1 hour)
2. Cryptography (4.5 hours)
   a. Basics (0.5)
   b. The Pre-Computer Age (0.5)
   c. Information Theoretic Cryptography (1)
   d. Private and Public Key Cryptography (1)
   e. Cryptographic Protocols (1)
   f. Summary and Systems on the Market (0.5)
3. Operating Systems Protection (5 hours)
   a. Basics (0.25)
   b. Attack and Defense (0.75)
   c. Protection Policy (1)
   d. Protection Models (1)
   e. Implementation Techniques (0.5)
   f. Standards (0.5)
   g. Examples (0.5)
   h. Database Protection (0.5 hour)
4. Network Protection (1 hour)
5. Fault Tolerant Computing and Safety Engineering (0.5 hour)
6. Physical Security (0.5 hour)
7. Protection Management (0.5 hour)
8. Legal Issues (1.5 hours)
9. Summary (0.5 hour)

Demonstration Support

The entire course can be supported by demonstration hardware, such as cipher wheels, radio equipment, and alarm systems. Because of the introductory level of the course, good demonstrations will provide a better understanding of the scope and nature of the material than in-depth mathematical analysis. Guest lecturers should be sought for demonstrations of specialized equipment and video tapes of appropriate demonstrations should be provided where local demonstration is infeasible. A limited number of hardware devices for demonstration purposes have been developed by the author for this and related courses. Requests for such materials may be made of the author.

Each phase of a course can be accompanied by small software prototypes that demonstrate the principles. The author has used a subset of these software prototypes in the past, and a list of these prototypes, along with additional suggestions, follows. Source code should be provided for all of the software in this course because it is fundamental to the understanding of the nature of protection systems, but in some cases, the dissemination of this material may present the student with the ability to attack systems. The author may be contacted for special requests.

This list is annotated with S for software, H for hardware, and A for audio-visual aides to indicate the nature of the demonstration materials required.

Cryptography.

- A set of cipher wheels (H)
- A crypto system for creating and breaking simple cipher systems (S)
- Simple information analysis tools (S)
- Data compression and decompression tools (S)
- DES and RSA cryptosystems (S)
- Samples of market systems (S,H)

Operating Systems Protection.

- Examples of attacks (at least one of each type mentioned) (S,H,A)
- A quality password generator and analyzer (S)
- An algorithmic authentication scheme (S)
- A prototype automated administrative assistant (S)
- A simple capability based system (S)
- A simple theorem prover (S)
- A simple database with statistical capabilities (S)
- Sample statistical and transitive attacks (S,A)
- A sample database with field encryption (S)

Network Protection.

- A key distribution system (A,S)
- A communications package with encryption and authentication (S)
- Network examples of selected attacks and defenses (A,H,S)

Fault Tolerant Computing and Safety Engineering.

- Example fatal and nonfatal accidents and histories (A)
- Example safety analysis tools (S)
- Example safety assurance tools (S,A)
- Examples from coding theory (S)
- Example hardware and software mechanisms (S,H)

Physical Security.

- A simple line tapper program (S,A)
- An example backup program and corresponding management plan (S,A)

Protection Management.

- A cost/benefit analysis system with a built in example (S)
- A configuration management analysis program (S)

Legal Issues.

- Samples of access to databases over telephones (S,H)
- Sample credit report generation and modification (S)

Exercises

The following exercises for students have been found useful in teaching this material. They are organized according to the module content outline.

Cryptography.

- Three simple ciphers are given to be broken.

Information Theoretic Cryptography.

- Some moderately complex ciphers are given to be broken (assistance program provided).

Private and Public Key Cryptography.

- Try out the DES and RSA (software provided).

Cryptographic Protocols.

- Perform a time space analysis of a sample hybrid system.

Operating System Protection.

- Write a covert channel program using disk space.
- Write a simple login spoofing program.
- Write a simple cryptographic checksum program.
- Write a simple Trojan horse program.
- Write a simple subject/object based system.
- Rewrite and prove the previous subject/object homework correct.
- Perform a few simple sample TSEC evaluations based on given data.
- Write a simple database with field encryption and subject/object protection.
- Demonstrate an aggregation attack on this system.
- Demonstrate a simple aggregation defense and its failing.

Network Protection.

- Write a simple TFC, FCP, and collusion analysis program.

Fault Tolerant Computing and Safety Engineering.

- Analyze relative risks for a set of potentially hazardous events.
- Write a database program that compensates for programming errors with redundant software structures.
- Analyze the likelihood of failure of a fault tolerant computer design.

Protection Management.

- Implement a simple cost/benefit optimization program.
- Solve a simple protection decision problem with the cost/benefit program.
- Write a simple failure analysis management database.

Legal Issues.

- Examine a legal case on information protection.

Bibliography

Adleman82
Adleman, L. On Breaking the Iterated Merkle-Hellman Public-Key Cryptosystem. *Advances in Cryptology*. Aug., 1982, 23-25.

This paper is one in a series that demonstrate methods by which the knapsack public key cryptosystem and a series of evolutions thereof may be broken. The clear indication of this series of papers is that a system must be fundamentally sound, and that the attempt to fix an unsound system via successive iterations on the same idea does not seem to help eliminate the fundamental unsoundness of it. The open discussion and exchanges in these papers also indicates clearly the advantages of publicly revealing classes of cryptosystems in terms of determining their ultimate security. We note that even after this series of papers was published, many manufacturers marketed products that used the knapsack system.

Anderson72
Anderson, J. *Computer Security Technology Planning Study*. ESD-TR-73-51, USAF Electronic Systems Division, Oct., 1972.

Bell73
Bell, D., and L. LaPadula. *Secure Computer Systems: Mathematical Foundations and Model*. The MITRE Corporation, 1973.

Benzel84
Benzel, T. Further Analysis of the SCOMP System Verification. *7th Security Conference*. DoD/NBS, Sept., 1984.

This paper is one in a series that examines the formal verification process used to certify the SCOMP operating system. This is a long and stringent mathematical process that is required to prove formally the protection properties of a system. It is noteworthy that SCOMP is certified A1 under the Orange Book criterion by the US DoD.

Biba77
Biba, K. *Integrity Considerations for Secure Computer Systems*. USAF Electronic Systems Division, 1977.

Branstad78
D. Branstad. Security of Computer Communications. *IEEE Communications* (Nov. 1978), 33-40.

Brinch-Hansen73
Brinch-Hansen, P. *Operating System Principles*. Prentice-Hall, Englewood Cliffs, N.J., 1973.

Chaum83a
Chaum, D. *Title unknown*. Ph.D. Th., Univ. Calif., Santa Barbara, 1983.

Chaum83b
Advances in Cryptology. D. Chaum, ed. Plenum Press, 1983.

This and the following two references are examples of the annual conference proceedings of the International Association of Cryptologic Research and are representative of the state of the art in cryptography and cryptographic research. They involve a great deal of mathematical rigor, and reveal the dramatic nature of cryptographic research and its changes over small periods of time.

Chaum84
Advances in Cryptology. D. Chaum, ed. Plenum Press, 1984.

Chaum85
Advances in Cryptology. D. Chaum, ed. Plenum Press, 1985.

Chen78a
Chen, L. *Improving Software Reliability by N-version Programming*. UCLA-ENG-7843, UCLA Computer Science Dept., 1978.

Chen78b
Chen, L. and A. Avizienis. N-version Programming: A Fault Tolerance Approach to Reliability of Software Operation. *Digest*. FTCS-8, June, 1978, 3-9.

Cohen84a
Cohen, F. Computer Security Methods and Systems. *1984 Conference on Information Systems and Science*. CISS, Princeton University, 1984.

Cohen84b
Cohen, F. Computer Viruses - Theory and Experiments. *7th Security Conference*. DoD/NBS, Sept., 1984.

This well known paper introduces the computer virus attack, and points out a number of unsolvable

problems in its prevention, detection, and removal. The class of attack introduced here is particularly important because of its transitive nature, and thus its ability to spread from system to system and become a global problem. This paper is the first paper on this topic that has in-depth coverage and significant results of theoretical interest. It has had a significant effect on the concern over integrity on the use of computer systems.

Cohen85
Cohen, F. A Secure Computer Network Design. *Computers and Security*. IFIP, March, 1985.

This paper introduces a class of network designs based on distributed domains which assures multilevel operation over secure and insecure communications lines. It includes analysis of potential attacks on the network, describes the effects and ramifications of covert channels and traffic analysis, and explores the ramifications of combining a security and integrity policy.

Cohen86
Cohen, F. Protection and Administration of Information Networks under Partial Orderings. *Computers and Security*. IFIP, Oct., 1986.

This paper examines the security, integrity, and lattice policies, and extends them to a more general partial ordering. The analysis of collusions of individuals and the effect of time on the control of information flow point out some complexities involved in policy making. The specification of automated administrative assistance is nearly unique in this context. The extensive use of examples throughout makes this paper particularly suitable for class presentation.

Cornwell84
Cornwell, M., and R. Jacob. Towards Multilevel-Secure Message Systems: Techniques Employed in Prototype Systems. *7th Computer Security Conference*. DoD/NBS, Sept., 1984.

Davies83
Davies, D. Use of the 'Signature Token' to Create a Negotiable. *Advances in Cryptology*. IACR, Aug., 1983, 377-382.

Davio83
Davio, M., Y. Desmedt, M. Fosseprez, R. Govaerts, J. Hulsbosch, P. Neutjens, P. Piret, J. Quisquater, J. Vandevalle, and P. Wouters. Analytical Characteristics of the DES. *Advances in Cryptology*. IACR, Aug., 1983, 171-202.

DeMillo83
DeMillo, R., and M. Merritt. Protocols for Data Security. *IEEE Computer 16*, 2 (Feb. 1983), 39-54.

Denning75
Denning, D. *Secure Information Flow in Computer Systems*. Ph.D. Th., Purdue University, West Lafayette, IN, May 1975.

Denning82
Denning, D. *Cryptography and Data Security*. Addison-Wesley, Reading, MA, 1982.

This text covers a great deal of the course material at a depth sufficient for a two semester course (when used with additional materials). It lacks coverage of relatively recent topics, physical security, social implications, and cost/benefit analysis, but serves as a useful reference with many good citations.

Denning83
Denning, D. Field Encryption and Authentication. *Advances in Cryptology*. Aug., 1984, 231-247.

Denning85a
Denning, D. Database Security. *International Data Security Conference*. Feb., 1985.

Denning85b
Denning, D. Commutative Filters for Reducing Inference Threats in Multilevel Database Systems. *Proceedings of the 1985 Symposium on Security and Privacy*. April, 1985, 134-146.

Diffie76
Diffie, W., and M. Hellman. New Directions in Cryptography. *IEEE Trans. Info. Theory IT-22*, 6 (Nov. 1976), 644-654.

This famous paper introduced the concept of public key cryptography and thus set a new direction for cryptography. Many new concepts and ideas were presented including the use of publicly verifiable digital signatures, elimination of the key distribution problem, the use of trapdoor transformations for cryptography, and the use of computational complexity for security rather than key secrecy.

Diffie77
Diffie, W., and M. Hellman. Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *IEEE Computer 10*, 6 (June 1977), 74-84.

Abstract: The National Bureau of Standards (NBS) has adopted a data encryption standard designed by

IBM. Unfortunately, the proposed standard is too weak for some applications and should be modified. This paper is intended to carefully set down reasoning so that the technical community can form its own opinion.

This paper details the exhaustive search attack against the DES, and shows that the system is not secure against a concerted attacker. This paper is a result of a meeting between DoD, NBS, and external experts in which the DES was discussed prior to its becoming a standard. The paper is quite revealing.

DOJ82

Department of Justice. *Computer Crime - Computer Security Techniques*. U.S. Department of Justice, Bureau of Justice Statistics, 1982. US GPO 1982-361-233/1873.

This report provides a broad coverage of computer security techniques, and a vast array of evaluation specifications that are useful in the analysis of computer systems for exposures. This is a broad and in-depth enumeration of techniques and analyses, and should be a useful resource to the instructor.

Feiertag79

Feiertag, R., and P. Neumann. The Foundations of a Provable Secure Operating System (PSOS). *National Computer Conference*. AFIPS, 1979, 329-334.

This paper describes the design of a secure operating system. It gives details of the design and analysis techniques as well as information on the structure of the system and what was involved in testing, implementation, and operation. This is particularly valuable for those interested in the design of operating systems protection.

Feistel

Feistel, H., W. Notz, and J. Smith. Some Cryptographic Techniques for Machine-to-Machine Data Communications. *Comm. ACM 18*, 11 (Nov. 1975), 1545-1554.

Fenton73

Fenton, J. *Information Protection Systems*. Ph.D. Th., U. of Cambridge, 1973.

Fernandez81

Fernandez, E., R. Summers, and C. Wood. *Database Security and Integrity*. Addison-Wesley, 1981.

This book provides in-depth coverage of database security and integrity. It cites most of the important papers and gives solid coverage in a clear and concise manner. Because progress in database security and integrity is relatively slow and tends to parallel operating system protection methods, little fundamental progress has been made in the last five years in this area. This text would be adequate as a text for a full course in database protection.

Gifford82

Gifford, D. Cryptographic Sealing for Information Secrecy and Authentication. *Comm. ACM 25*, 4 (April 1982), 274-285.

Abstract: A new protection mechanism is described that provides general primitives for protection and authentication. The mechanism is based on the idea of sealing an object with a key. Sealed objects are self-authenticating, and in the absence of an appropriate set of keys, only provide information about the size of their contents. New keys can be freely created at any time, and keys can also be derived from existing keys with operators that include Key-And and Key-Or. This flexibility allows the protection mechanism to implement common protection mechanisms such as capabilities, access control lists, and information flow control. The mechanism is enforced with a synthesis of conventional cryptography, public-key cryptography, and a threshold scheme.

Gold79

Gold, B., R. Linde, R. Peeler, M. Schaefer, J. Scheid, and P. Ward. A Security Retrofit of VM/370. *National Computer Conference*. AFIPS, 1979, 335-344.

Harrison76

Harrison, M., W. Ruzzo, and J. Ullman. Protection in Operating Systems. *Proceedings of the ACM*. ACM, 1976.

This famous paper lays the groundwork for the design and mathematical analysis of protection systems. It points out several severe problems with an extremely general class of protection systems, and demonstrates that even simple protection systems may have complex ramifications. The subject/object matrix is introduced as a method for understanding protection systems, and it is noteworthy that this matrix is also used in implementation of secure systems.

Highland84

Highland, H. Case Study of Market Cryptosytstems for PCs. *IFIP-SEC84*. IFIP, July, 1984.

This paper presents a case study of several cryptosystems available for the protection of information on microcomputers. The products on the market represent a wide range of security levels, but

none of them is extremely strong, and all but a few defend only against relatively unsophisticated attackers.

Hoffman82
Hoffman, L. Impacts of Information System Vulnerabilities on Society. *National Computer Conference*. AFIPS, 1982, 461-467.

Kam79
Kam, J., and G. Davida. Structured Design of Substitution-Permutation Encryption Networks. *IEEE Trans. Computers C-28*, 10 (Oct. 1979).

This paper details the derivation of tables in the DES and is useful for understanding the nature of the confusion and diffusion transforms as substitution and permutation networks in the system, and reveals underlying design motivations.

Kelly83
Kelly, J. and A. Avizienis. A Specification Oriented Multi-Version Software Experiment. *Symposium on Fault Tolerant Computing*. IEEE, 1983, 120-126.

Klein83
Klein, M. *Department of Defense Trusted Computer System Evaluation Criteria*. Department of Defense, Fort Meade, Md. 20755, 1983.

This specification, commonly called the "Orange Book" because of the color of the cover, specifies the criterion by which the US military, and de facto, the rest of the government and industry, evaluate operating system protection mechanisms. The document was formed as the result of many years of work by a large group of internationally known experts in the security fields, and comprises a consolidation of the state of the art in understanding the criterion by which such systems should be evaluated as of 1983. A number of classes of security are defined in terms of their ability to prevent attack, and the degree to which they can be demonstrated to do so. The concept of "multilevel" security is introduced, and a number of good references are provided.

Knight78
Knight, H. Cryptanalyst's Corner. *Cryptologia 2*, 1 (Jan. 1978), 68-74.

This paper details a classification scheme for ciphers and cryptosystems, and details some of the statistical techniques used in breaking the "unknown" cipher. An entire series of articles follow this one, and may be of interest for individual classes or students.

Lampson73
Lampson, B. A Note on the Confinement Problem. *Comm. ACM 16*, 10 (Oct. 1973), 613-615.

Abstract: This note explores the problem of confining a program during its execution so that it cannot transmit information to any other program except its caller. A set of examples attempts to stake out the boundaries of the problem. Necessary conditions for a solution are stated and informally justified.

This famous paper introduces the "confinement problem," which appears to be an unsolvable security problem involved in the use of a program as a service. Many papers have cited this short paper, and its ramifications are widespread.

Landwehr83
Landwehr, C. The Best Available Technologies for Computer Security. *IEEE Computer 16*, 7 (July 1983).

Abstract: This concise overview of secure system developments summarizes past and current projects, deciphers computer security lingo, and offers advice to prospective designers.

Linde75
Linde, R. Operating System Penetration. *National Computer Conference*. AFIPS, 1975, 361-368.

This paper introduces a wide variety of computer protection vulnerabilities that computer systems have long been vulnerable to. Many of these attacks have been further clarified by theoretical models, but the presentation makes it clear that ad hoc protection mechanisms will never suffice, and that a theoretically sound protection is required if we are to trust a system.

Littlewood84
Littlewood, B., and P. Keiller. Adaptive Software Reliability Modelling. *Symposium on Fault Tolerant Computing*. IEEE, 1984, 108-113.

McCauley79
McCauley, E. and P. Drongowski. KSOS - The Design of a Secure Operating System. *National Computer Conference*. AFIPS, 1979, 345-353.

This paper describes the design of a secure operating system. It gives details of the design and analysis techniques as well as information on the structure of the system and what was involved in testing, implementation, and operation. This is particularly valuable for those interested in the design of operating systems protection.

Merkle80
Merkle, R. Protocols for Public Key Systems. *Symposium on Security and Privacy.* IEEE, 1980.

Moore56
Moore, E., and C. Shannon. Reliable Circuits Using Less Reliable Relays. *J. Franklin Inst.* , 262 (Sept. 1956), 191-208.

NBS77
National Bureau of Standards. *Data Encryption Standard.* US Govt. Printing Office, 1977. FIPS PUB 46.

This document is the standard released from the National Bureau of Standards (NBS) that formally defines the Data Encryption Standard (DES) cryptosystem, which is probably the most widely used cryptosystem in the world at present.

Needham78
Needham, R., and M. Schroeder. Using Encryption for Authentication in a Large Network of Computers. *Comm. ACM 21*, 12 (Dec. 1978), 993-999.

Abstract: Use of encryption to achieve authenticated communication in computer networks is discussed. Example protocols are presented for the establishment of authenticated connections, for the management of authenticated mail, and for signature verification and document integrity guarantee. Both conventional and public-key encryption algorithms are considered as the basis for protocols.

Popek75
Popek, G., and C. Kline. A Verifiable Protection System. *Reliable Software Design.* IEEE, 1975, 294-304.

Popek79
Popek, G., M. Kampe, C. Kline, A. Stoughton, M. Urban, and E. Walton. UCLA Secure UNIX. *National Computer Conference.* AFIPS, 1979, 355-364.

This paper describes the design of a secure operating systems. It gives details of the design and analysis techniques as well as information on the structure of the system and what was involved in testing, implementation, and operation. This is particularly valuable for those interested in the design of operating systems protection.

Randell75
Randell, B. System Structure for Software Fault Tolerance. *IEEE Trans. Software Eng. SE-1*, 2 (June 1975), 220-223.

Abstract: This paper presents and discusses the rationale behind a method for structuring complex computing systems by the use of what we term "recovery blocks," "conversations," and "fault-tolerant interfaces." The aim is to facilitate the provision of dependable error detection and recovery facilities which can cope with errors caused by residual design inadequacies, particularly in the system software, rather than merely the occasional malfunctioning of hardware components.

Rivest78
Rivest, R., A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Comm. ACM 21*, 2 (Feb. 1978).

*Abstract: An encryption method is presented with the novel property that publicly revealing an encryption key does not thereby reveal the corresponding decryption key. This has two important consequences: (1) Couriers or other secure means are not needed to transmit keys, since a message can be enciphered using an encryption key publicly revealed by the intended recipient. Only he can decipher the message, since only he knows the corresponding decryption key. (2) A message can be "signed" using a privately held decryption key. Anyone can verify this signature using the corresponding publicly revealed encryption key. Signatures cannot be forged, and a signer cannot later deny the validity of his signature. This has obvious applications in "electronic mail" and "electronic funds transfer" systems. A message is encrypted by representing it as a number M, raising M to a publicly specified power e, and then taking the remainder when the result is divided by the publicly specified product, n, of two large secret prime numbers p and q. Decryption is similar; only a different, secret, power d is used, where e*d = 1 (mod (p-1)*(q-1)). The security of the system rests in part on the difficulty of factoring the published divisor, n.*

This famous paper is the basis for every presently unbroken public key cryptosystem. An encryption method is presented with the novel property that publicly revealing an encryption key does not thereby reveal the corresponding decryption key. This has two important consequences: (1) encryption keys may be distributed in the clear and (2) electronic signatures by the owner of the decryption key may be publicly verified, but not forged, and not denied by the signer. The security of the system is based on the difficulty of factoring the product of two large primes, a longstanding mathematical problem that appears to hold sufficient complexity as to allow breaking the system to be sufficiently complex. A breakthrough in the generation of large primes allows the complexity of generating these

primes to be made sufficiently fast as to be practical.

Scott84
Scott, R., J. Gault, D. McAllister, and J. Wiggs. Experimental Validation of Six Fault Tolerant Software Reliability Models. *Symposium on Fault Tolerant Computing*. IEEE, 1984, 102-107.

Shamir79
Shamir, A. How to Share a Secret. *Comm. ACM 22*, 11 (Nov. 1979), 612-613.

Abstract: In this paper we show how to divide data D *into* n *pieces in such a way that* D *is easily reconstructable from any* k *pieces, but even complete knowledge of* k-1 *pieces reveals absolutely no information about* D. *This technique enables the construction of robust key management schemes for cryptographic systems that can function securely and reliably even when misfortunes destroy half the pieces and security breaches expose all but one of the remaining pieces.*

Shannon48
Shannon, C. A Mathematical Theory of Communications. *Bell Systems Tech. J. 27*, 3 (July 1948).

Shannon49
Shannon, C. Communications Theory of Secrecy Systems. *Bell Systems Tech. J. 28* (1949), 656-715.

Shoch82
Shoch, J., and J. Hupp. The 'Worm' Programs - Early Experience with a Distributed Computation. *Comm. ACM 25*, 3 (March 1982), 172-180.

Abstract: The "worm" programs were an experiment in the development of distributed computations: programs that span machine boundaries and also replicate themselves in idle machines. A "worm" is composed of multiple "segments," each running on a different machine. The underlying worm maintenance mechanisms are responsible for maintaining the worm-finding free machines when needed and replicating the program for each additional segment. These techniques were successfully used to support several real applications, ranging from a simple multimachine test program to a more sophisticated real-time animation system harnessing multiple machines.

This paper reports on the accidental security ramifications of the Xerox worm program. This program was designed to do multiprocessing on a network of processors that could be taken over at any time for use as personal workstations. A bit error accidentally caused the program to refuse to give up control over the workstation, and because of the manner in which it spread, rebooting the workstation only reloaded the worm. Eventually most of the entire nationwide network had to be shut down, and a restart of the network from backups at a central cite had to be used.

Siewiorek82
Siewiorek, D., and R. Swarz. *The Theory and Practice of Reliable System Design*. Digital Press, Bedford, MA, 1982.

Simmons81
Simmons, G. Verification of the Nuclear Test Ban Treaty. *Oakland Conference on Computer Security*. IEEE, Aug., 1981.

This paper describes the use of the RSA cryptosystem in conjunction with cryptographic protocols to solve the problem of verification of the nuclear test ban treaty. Many interesting problems are presented and the solution actually used in this case is presented for world wide inspection.

Suzuki75
Suzuki, N. Verifying Programs by Algebraic and Logical Reduction. *Conference on Reliable Software*. IEEE, 1975, 473-481.

Thompson84
Thompson, K. Reflections on Trusting Trust. *Comm. ACM 27*, 8 (Aug. 1984), 761-763.

Abstract: To what extent should one trust a statement that a program is free of Trojan horses? Perhaps it is more important to trust the people who wrote the software.

Tompkins84
Tompkins, J. *Report on Computer Crime*. American Bar Association, 1984.

This is a well known report on computer crime from the Criminal Justice Section of the American Bar Association. It details a case study of computer crime among many segments of the society, including business, government, education, and the public at large. It is perhaps the most detailed and rigorous study of its kind, and it presents conclusions that are eye-opening in terms of the scope and degree of the computer crime problem.

Voelcker86
Voelcker, J., and P. Wallich. How Disks are 'Padlocked'. *IEEE Spectrum 23*, 6 (June 1986), 32-40.

This paper provides broad coverage of legal aspects

involved with software licensing and protection of floppy disks. It is interesting reading, requires little time and effort, and is suitable for increasing awareness.

vonNeumann56
von Neumann, J. *Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components.* Princeton University, 1956.

Walker84
Walker, S., P. Baker, J. Anderson, and S. Brand. Introduction to Network Security Evaluation. *7th Security Conference.* DoD/NBS, Sept., 1984.

Woodward79
Woodward, J. Applications for Multilevel Secure Operating Systems. *National Computer Conference.* AFIPS, 1979, 319-328.

Yau75
Yau, S., and R. Cheung. Design of Self Checking Software. *Conference on Reliable Software.* IEEE, 1975, 450-457.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS NONE
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT APPROVED FOR PUBLIC RELEASE DISTRIBUTION UNLIMITED
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) SEI-CM-5-1.2	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION SOFTWARE ENGINEERING INST. / 6b. OFFICE SYMBOL (If applicable) SEI	7a. NAME OF MONITORING ORGANIZATION SEI JOINT PROGRAM OFFICE
6c. ADDRESS (City, State and ZIP Code) CARNEGIE MELLON UNIVERSITY PITTSBURGH, PA 15213	7b. ADDRESS (City, State and ZIP Code) ESD/AVS HANSCOM AIR FORCE BASE, MA 01731
8a. NAME OF FUNDING/SPONSORING ORGANIZATION SEI JOINT PROGRAM OFFICE / 8b. OFFICE SYMBOL (If applicable) ESD/ AVS	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F1962890C0003
8c. ADDRESS (City, State and ZIP Code) CARNEGIE MELLON UNIVERSITY PITTSBURGH, PA 15213	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO.: 63752F / PROJECT NO.: N/A / TASK NO.: N/A / WORK UNIT NO.: N/A
11. TITLE (Include Security Classification) Information Protection	
12. PERSONAL AUTHOR(S) Fred Cohen, Lehigh University	
13a. TYPE OF REPORT FINAL / 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day) July 1987 / 15. PAGE COUNT 20
16. SUPPLEMENTARY NOTATION	

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB GR	information protection, operating system protection, security, network protection, cryptography

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

This module is a braod based introduction to information protection techniques. Topics include the history and present state of cryptography, operating system protection fault tolerance and safety engineering, physical protection techniques, management tradeoffs, legal issues, and current research trends. The successful student in this course will be prepared for an in-depth course in any of these topics.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ OTIC USERS ☒	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED, UNLIMITED DISTRIBUTION
22a. NAME OF RESPONSIBLE INDIVIDUAL JOHN S. HERMAN, Capt, USAF	22b. TELEPHONE NUMBER (Include Area Code) 412 268-7630 / 22c. OFFICE SYMBOL ESD/AVS (SEI JPO)

The Software Engineering Institute (SEI) is a federally funded research and development center, operated by Carnegie Mellon University under contract with the United States Department of Defense.

The SEI Software Engineering Curriculum Project is developing a wide range of materials to support software engineering education. A *curriculum module* (CM) identifies and outlines the content of a specific topic area, and is intended to be used by an instructor in designing a course. A *support materials* package (SM) contains materials related to a module that may be helpful in teaching a course. An *educational materials* package (EM) contains other materials not necessarily related to a curriculum module. Other publications include software engineering curriculum recommendations and course designs.

SEI educational materials are being made available to educators throughout the academic, industrial, and government communities. The use of these materials in a course does not in any way constitute an endorsement of the course by the SEI, by Carnegie Mellon University, or by the United States government.

Comments on SEI educational materials and requests for additional information should be addressed to the Software Engineering Curriculum Project, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213. Electronic mail can be sent to education@sei.cmu.edu on the Internet.

Curriculum Modules (* Support Materials available)

CM-1 [superseded by CM-19]
CM-2 Introduction to Software Design
CM-3 The Software Technical Review Process*
CM-4 Software Configuration Management*
CM-5 Information Protection
CM-6 Software Safety
CM-7 Assurance of Software Quality
CM-8 Formal Specification of Software*
CM-9 Unit Testing and Analysis
CM-10 Models of Software Evolution: Life Cycle and Process
CM-11 Software Specifications: A Framework
CM-12 Software Metrics
CM-13 Introduction to Software Verification and Validation
CM-14 Intellectual Property Protection for Software
CM-15 Software Development and Licensing Contracts
CM-16 Software Development Using VDM
CM-17 User Interface Development*
CM-18 [superseded by CM-23]
CM-19 Software Requirements
CM-20 Formal Verification of Programs
CM-21 Software Project Management
CM-22 Software Design Methods for Real-Time Systems*
CM-23 Technical Writing for Software Engineers
CM-24 Concepts of Concurrent Programming
CM-25 Language and System Support for Concurrent Programming*
CM-26 Understanding Program Dependencies

Educational Materials

EM-1 Software Maintenance Exercises for a Software Engineering Project Course
EM-2 APSE Interactive Monitor: An Artifact for Software Engineering Education
EM-3 Reading Computer Programs: Instructor's Guide and Exercises